```
class Node:
    def __init__(self, freq_, symbol_, left_=None, right_=None):
        self.freq = freq_
        self.symbol = symbol_
        self.left = left_
        self.right = right_
        self.huff = ""
def print_nodes(node, val=""):
    new_val = val + str(node.huff)
    if node.left:
     print_nodes(node.left, new_val)
    if node.right:
        print_nodes(node.right, new_val)
    if not node.left and not node.right:
        print(f"{node.symbol} -> {new_val}")
chars = ["a", "b", "c", "d", "e", "f"]
freq = [10, 4, 9, 7, 1, 15]
nodes = [Node(freq[x], chars[x]) for x in range(len(chars))]
while len(nodes) > 1:
    nodes = sorted(nodes, key=lambda x: x.freq)
    left = nodes[0]
    right = nodes[1]
    left.huff = 0
    right.huff = 1
    newNode = Node(left.freq + right.freq, left.symbol + right.symbol, left, right)
    nodes.remove(left)
    nodes.remove(right)
    nodes.append(newNode)
print("Characters :", f'[{", ".join(chars)}]')
print("Frequency :", freq, "\n\nHuffman Encoding:")
print_nodes(nodes[0])
```

OUTPUT:

Characters : [a, b, c, d, e, f]

Frequency : [10, 4, 9, 7, 1, 15]

Huffman Encoding:

c -> 00

a -> 01

e -> 1000

b -> 1001

d -> 101

f -> 11